CHAINSECURITY

PUBLIC

Code Assessment
of the PAS Timelock Calldata Generator Smart Contracts

August 05, 2026

Produced for

Sky

Contents

1 Executive Summary

Dear Sky Team,

Thank you for trusting us to help Sky with this security audit. Our executive summary provides an overview of subjects covered in our audit of the latest reviewed contracts of PAS Timelock Calldata Generator according to **Scope** to support you in forming an opinion on their security risks.

Sky implements a calldata generator for its Prime Allocation System (PAS). `TimelockCalldataGenerator` is a stateless, `pure` helper that encodes the payloads for PAS governance operations, both direct `BeamState` configuration and PAU controller actions, so they can be scheduled through the PAS `Timelock`.

The most critical subject covered in our audit is the functional correctness of the produced calldata. The general subjects covered are code complexity and documentation.

Security regarding functional correctness is high.

In summary, we find that the codebase provides a high level of security.

It is important to note that security audits are time-boxed and cannot uncover all vulnerabilities. They complement but don't replace other vital measures to secure a project.

The following sections will give an overview of the system, our methodology, the issues uncovered, and how they have been addressed. We are happy to receive questions and feedback to improve our service.

Sincerely yours,
ChainSecurity

2 Assessment Overview

In this section, we briefly describe the overall structure and scope of the engagement, including the code commit which is referenced throughout this report.

2.1 Scope

The assessment was performed on the source code files inside the PAS Timelock Calldata Generator repository based on the documentation files. The table below indicates the code versions relevant to this report and when they were received.

Version	Date	Commit Hash	Note
1	4 August 2026	848ab88d21838d5edd6281be8840dce92b5bf9a4	Initial Review

For the Solidity smart contracts, the compiler version `0.8.34` was chosen.

The following file was in scope:

```
src/
└── TimelockCalldataGenerator.sol
```

2.1.1 Excluded from scope

Everything outside the file listed above is out of scope. In particular, the PAS and PAU contracts that the generated calldata targets are only referenced for context and were not reviewed here:

- The PAS contracts (`Timelock`, `BeamState`, `Configurator`, `AccessControls`).
- The Diamond PAU controllers and facets.

These dependencies are assumed to behave as specified.

3 System Overview

This system overview describes the initially received version (**Version 1**) of the contracts as defined in the Assessment Overview.

Furthermore, in the findings section, we have added a version icon to each of the findings to increase the readability of the report.

PAS (sky-ecosystem/pas) is Sky's Prime Allocation System. It is the governance layer where the `Timelock`, `BeamState`, and `Configurator` schedule and apply configuration for the diamond-pau controllers. Its privileged operations are submitted on-chain as ABI-encoded calldata to the PAS `Timelock`.

`TimelockCalldataGenerator` is the single contract in scope and is a stateless data encoder for those operations. It holds no funds, has no privileged roles, and never changes state. Every function is `pure` and returns the ABI-encoded `bytes` payload. This gives proposers one place to build the calldata they would otherwise write by hand.

The generator produces two kinds of payload:

- **Direct `BeamState` config calls.** Encode a `BeamState` function directly.
- **PAU actions.** Encode an inner call (a Diamond PAU controller action or an `AccessControls` role change) and wrap it via `BeamState.addInitControllerActions(data, controller)` for later execution through the Configurator/cBEAM path.

Both are used the same way. Call a per-action encoder to get an individual `BeamState` payload, then pass a set of those payloads to `scheduleBatch`, which wraps them into `Timelock.scheduleBatch(...)` calldata (all targets set to `BeamState`, all values zero).

4 Trust Model

`TimelockCalldataGenerator` has no access control, no state, and no funds. Every function is `pure` and only returns ABI-encoded `bytes`. It therefore has no on-chain roles of its own. The only actor is the caller, and all privileged behaviour lives in the downstream PAS contracts that the calldata targets.

Because the generator validates nothing, we assume governance reviews and validates the payloads before submitting them, and that all access control and state changes happen in the downstream PAS contracts the calldata targets.

5 Terminology

For the purpose of this assessment, we adopt the following terminology. To classify the severity of our findings, we determine the likelihood and impact (according to the CVSS risk rating methodology).

- **Likelihood** represents the likelihood of a finding to be triggered or exploited in practice
- **Impact** specifies the technical and business-related consequences of a finding
- **Severity** is derived based on the likelihood and the impact

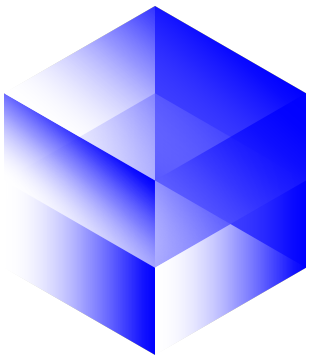
We categorize the findings into four distinct categories, depending on their severity. These severities are derived from the likelihood and the impact using the following table, following a standard risk assessment procedure.

Likelihood	**Impact**		
	High	Medium	Low
High	**Critical**	**High**	**Medium**
Medium	**High**	**Medium**	**Low**
Low	**Medium**	**Low**	**Low**

As seen in the table above, findings that have both a high likelihood and a high impact are classified as critical. Intuitively, such findings are likely to be triggered and cause significant disruption. Overall, the severity correlates with the associated risk. However, every finding's risk should always be closely checked, regardless of severity.

6 Findings

In this section, we describe the findings identified during the course of the engagement. The findings are categorized by severity as explained in the Terminology section. Below we provide an overview of the total number of findings per severity, as well as the number of open issues.



No Open Issues

Total number of findings

Critical	High	Medium	Low	Informational
0	0	0	0	0

7 Limitations and use of report

Security assessments cannot uncover all existing vulnerabilities; even an assessment in which no vulnerabilities are found is not a guarantee of a secure system. However, code assessments enable the discovery of vulnerabilities that were overlooked during development and areas where additional security measures are necessary. In most cases, applications are either fully protected against a certain type of attack, or they are completely unprotected against it. Some of the issues may affect the entire application, while some lack protection only in certain areas. This is why we carry out a source code assessment aimed at determining all locations that need to be fixed. Within the customer-determined time frame, ChainSecurity has performed an assessment in order to discover as many vulnerabilities as possible.

The focus of our assessment was limited to the code parts defined in the engagement letter. We assessed whether the project follows the provided specifications. These assessments are based on the provided threat model and trust assumptions. We draw attention to the fact that due to inherent limitations in any software development process and software product, an inherent risk exists that even major failures or malfunctions can remain undetected. Further uncertainties exist in any software product or application used during the development, which itself cannot be free from any error or failures. These preconditions can have an impact on the system's code and/or functions and/or operation. We did not assess the underlying third-party infrastructure which adds further inherent risks as we rely on the correct execution of the included third-party technology stack itself. Report readers should also take into account that over the life cycle of any software, changes to the product itself or to the environment in which it is operated can have an impact leading to operational behaviors other than those initially determined in the business specification.